COM 3110 Semester 1 Project Report

# Implementation

All three schemes are performed through the forQuery of the **Retrieve** class. A **Retrieve** object is constructed from the variable **index** and the variable **termWeighting**. The variable **index** is a dictionary where each key is a word and each value is a dictionary. In each of the sub-dictionaries, each key is a docid from documents.txt and each value is the frequency of the word in the document with said docid. The variable **termWeighting** determines whether binary, tf or tfidf is used.

All three schemes use the variable **query**, which is a dictionary where the keys are strings. First they filter out the sub-dictionaries in index that have a word in the current query. Then they calculate the scores of each word for different docids, the calculation is different depending on the scheme used. The scores are summed up according the docid. The docids are then sorted by highest to lowest score and returned.

## Binary

Increment the document’s score if the word is in it.

## Term Frequency (tf)

Words that appear frequently in one document score higher. To calculate this, we first calculate the number of distinct words for each document by looping through the sub-dictionaries in the **index** variable and incrementing the word count for a document every time its docid appears as a key.

## Term Frequency Inverse Document Frequency (tfidf)

Just like in **tf**, terms that appear frequently in one document score higher. Unlike **tf**, these are also multiplied by the Inverse Document Frequency (**idf**) which is given by the formula:

Where is the term and is the set of all documents. [1]

This is so that words that appear in many documents score lower.

# Performance

I’ve timed each run three times on my own laptop (Time1) and Diamond PCs (Time2), measured in minutes and seconds.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| binary | nsns | nsws | wsns | wsws |
| Time1 | 8.067734718322754 | 5.635002851486206 | 4.880497455596924 | 3.2782843112945557 |
| Time2 | 7.370153188705444 | 4.605186223983765 | 4.5701844692230225 | 3.1341757774353027 |
| Rel\_Retr | 84 | 91 | 107 | 116 |
| Precision | 0.13 | 0.14 | 0.17 | 0.18 |
| Recall | 0.11 | 0.11 | 0.13 | 0.15 |
| F-measure | 0.12 | 0.13 | 0.15 | 0.16 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| tf | nsns | nsws | wsns | wsws |
| Time1 | 09:48.2148609161377 | 10:59.0522611141205 | 49.985572814941406 | 01:49.94113659858704 |
| Time2 | 6:20.1829800605774 | 7:06.86233258247375 | 32.81787657737732 | 70.18922328948975 |
| Rel\_Retr | 27 | 41 | 70 | 81 |
| Precision | 0.04 | 0.06 | 0.11 | 0.13 |
| Recall | 0.03 | 0.05 | 0.09 | 0.10 |
| F-measure | 0.04 | 0.06 | 0.10 | 0.11 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| tfidf | nsns | nsws | wsns | wsws |
| Time1 | 09:39.91002202034 | 10:52.1373827457428 | 49.847440242767334 | 01:48.46404767036438 |
| Time2 | 06:11.56094694137573 | 06:57.80731892585754 | 33.019713163375854 | 01:09.03871083259583 |
| Rel\_Retr | 83 | 123 | 89 | 133 |
| Precision | 0.13 | 0.19 | 0.14 | 0.21 |
| Recall | 0.10 | 0.15 | 0.11 | 0.17 |
| F-measure | 0.12 | 0.17 | 0.12 | 0.19 |

Binary is the fastest whereas tfidf is slowest. Tf is the worst performing. Binary has the best performance when there isn’t stemming or a stoplist but with those options, especially stemming, it is possible for tfidf to perform better. Having a stoplist and stemming can improve performance.

# References

|  |  |
| --- | --- |
| [1] | P. Senin, “Term Frequency - Inverse Document Frequency statistics,” 27 10 2016. [Online]. Available: https://jmotif.github.io/sax-vsm\_site/morea/algorithm/TFIDF.html. [Accessed 06 11 2017]. |